Applied Mathematical Sciences, Vol. 7, 2013, no. 97, 4813 - 4819
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ams.2013.36321

The Shortest Common Ordered Supersequence Problem

Anna Gorbenko

Department of Intelligent Systems and Robotics
Ural Federal University
620083 Ekaterinburg, Russia
gorbenko.ann@gmail.com

Abstract

In this paper, we consider the problem of the shortest common ordered supersequence. In particular, we consider an explicit reduction from the problem to the satisfiability problem.

Keywords: ordered supersequence, satisfiability, **NP**-complete

It is well-known that investigation of different regularities can be used to retrieve various important knowledge (see e.g. [1] – [6]). In particular, different variants of the shortest common supersequence problem play important roles in data compression and DNA sequencing. In this paper, we consider the shortest common ordered supersequence problem.

Let $\Sigma = \{a_1, a_2, \ldots, a_m\}$ be a finite alphabet of letters. We assume that Σ^+ is the set of all nonempty strings over Σ. Let $\mathcal{S} = \{S_i \mid 1 \leq i \leq n, S_i \in \Sigma^+\}$. We assume that the length of a string S is the number of letters in it and is denoted as $|S|$. We use $S[i]$ to denote the ith letter in string S, and $S[i, j]$ to denote the substring of S consisting of the ith letter through the jth letter. For given two strings S and T over Σ, the string S is a subsequence of T if there is $F_S : \{1, 2, \ldots, |S|\} \to \{1, 2, \ldots, |T|\}$ such that $F_S(i) < F_S(j)$ and $S[l] = T[F_S(l)]$, for all $1 \leq i < j \leq |S|$ and $1 \leq l \leq |S|$. We will say that the string T is an ordered supersequence of $\mathcal{S}$ if S_i is a subsequence of T, for all $1 \leq i \leq n$, and there are $F_{S_1}, F_{S_2}, \ldots, F_{S_n}$ and a permutation σ such that

if $i < j$, then $F_{S_{\sigma(i)}}(1) < F_{S_{\sigma(j)}}(1)$ and $F_{S_{\sigma(i)}}(|S_{\sigma(i)}|) < F_{S_{\sigma(j)}}(|S_{\sigma(j)}|)$, for all $i, j \in \{1, 2, \ldots, n\}$.

The Shortest Common Ordered Supersequence Problem (SCOS):

Instance: *A collection $\mathcal{S}$ of strings over Σ and a positive integer k.*

Question: *Is there a string S such that $|S| \leq k$ and S is an ordered supersequence of $\mathcal{S}$?*

Theorem. SCOS *is* **NP***-complete..*

Proof. It is easy to see that SCOS in **NP**. Let $\mathcal{S}$ is a set of strings over Σ. Let $\Pi = \{b_1, b_2, \ldots, b_{2n}\}$. Let $\mathcal{T} = \{b_i S_i b_{n+i} \mid 1 \leq i \leq n\}$. By definition of ordered supersequence, if T is an ordered supersequence of $\mathcal{T}$, then there is a permutation σ such that

$$T = T_1 b_{\sigma(1)} T_2 b_{\sigma(2)} \ldots T_n b_{\sigma(n)} T_{n+1} b_{n+\sigma(1)} T_{n+2} b_{n+\sigma(2)} \ldots T_{2n} b_{n+\sigma(n)} T_{2n+1}$$

and $T_2 \ldots T_n T_{n+1} T_{n+2} \ldots T_{2n}$ is a supersequence of $\mathcal{S}$. Therefore, there is a string T such that $|T| \leq k$ and T is an ordered supersequence of $\mathcal{T}$ if and only if there is a string S such that $|S| \leq k - 2n$ and S is a supersequence of $\mathcal{S}$. Since the shortest common supersequence problem is **NP**-complete (see e.g. [7]), it is clear that SCOS is **NP**-hard. □

Since SCOS is **NP**-complete, we need some efficient algorithm to solve the problem. Note that encoding various hard problems as instances of different variants of the satisfiability problem and solving them with efficient satisfiability algorithms has caused considerable interest recently (see e.g. [8] – [16]). In this paper, we consider an explicit reduction from SCOS to the satisfiability problem.

Let

$$\begin{aligned}
\varphi[1] &= \wedge_{1\leq i\leq k} \vee_{1\leq j\leq m} x[i,j], \\
\varphi[2] &= \wedge_{1\leq i\leq k} \wedge_{1\leq j[1]<j[2]\leq m} (\neg x[i,j[1]] \vee \neg x[i,j[2]]), \\
\varphi[3] &= \wedge_{1\leq i\leq n} \wedge_{1\leq j\leq |S_i|} \vee_{1\leq s\leq k} y[i,j,s], \\
\varphi[4] &= \wedge_{1\leq i\leq n} \wedge_{1\leq j\leq |S_i|} \wedge_{1\leq s[1]<s[2]\leq k} (\neg y[i,j,s[1]] \vee \neg y[i,j,s[2]]), \\
\varphi[5] &= \wedge_{1\leq i\leq n} \wedge_{1\leq j[1]<j[2]\leq |S_i|} \wedge_{1\leq s[2]\leq s[1]\leq k} (\neg y[i,j[1],s[1]] \vee \\
&\quad \neg y[i,j[2],s[2]]), \\
\varphi[6] &= \wedge_{1\leq i\leq n} \wedge_{1\leq j\leq |S_i|} \wedge_{1\leq s\leq k} \wedge_{\substack{1\leq t\leq m, \\ S_i[j]\neq a_t}} (\neg y[i,j,s] \vee \neg x[s,t]), \\
\varphi[7] &= \wedge_{1\leq i\leq n} \vee_{1\leq j\leq n} z[i,j], \\
\varphi[8] &= \wedge_{1\leq i\leq n} \wedge_{1\leq j[1]<j[2]\leq n} (\neg z[i,j[1]] \vee \neg z[i,j[2]]), \\
\varphi[9] &= \wedge_{1\leq i[1]<i[2]\leq n} \wedge_{1\leq j\leq n} (\neg z[i[1],j] \vee \neg z[i[2],j]), \\
\varphi[10] &= \wedge_{1\leq i[1]\leq n} \wedge_{1\leq i[2]\leq n} \wedge_{1\leq j[1]<j[2]\leq n} \wedge_{\substack{1\leq s[2]\leq s[1]\leq k, \\ 1\leq t[2]\leq t[1]\leq k}} (\neg z[i[1],j[1]] \vee
\end{aligned}$$

$$\neg z[i[2], j[2]] \vee \neg y[i[1], 1, s[1]] \vee \neg y[i[2], 1, s[2]] \vee$$
$$\neg y[i[1], |S_{i[1]}|, t[1]] \vee \neg y[i[2], |S_{i[2]}|, t[2]]),$$
$$\xi = \wedge_{i=1}^{10} \varphi[i].$$

It is easy to check that there is a string S such that $|S| \leq k$ and S is an ordered supersequence of $\mathcal{S}$ if and only if ξ is satisfiable. It is clear that ξ is a CNF. Using standard transformations we can obtain an explicit transformation ξ into ζ such that $\xi \Leftrightarrow \zeta$ and ζ is a 3-CNF. Clearly, ζ gives us an explicit reduction from SCOS to 3SAT.

To obtain optimal solutions of SCOS we use genetic algorithms OA[1] (see [17]), OA[2] (see [18]), and OA[3] (see [11]) for the satisfiability problem. We have used heterogeneous cluster. Each test was runned on a cluster of at least 100 nodes. Note that due to restrictions on computation time (20 hours) we used savepoints. Selected experimental results are given in Tab. 1.

time	average	max	best
OA[1]	6.48 hr	21.59 hr	8 min
OA[2]	7.14 hr	24.18 hr	4.2 min
OA[3]	4.37 hr	16.15 hr	3.1 min

Table 1: Experimental results for OA[1], OA[2], and OA[3].

Now, we consider some genetic algorithms for SCOS. At first, we consider a relatively standard genetic algorithm GA[1] for solution of SCOS. Let $\mathbf{S}$ be a collection of strings over Σ. Let $\mathcal{S}_n$ be the symmetric group over $\{1, 2, \ldots, n\}$. We assume that

$$\mathbf{T} = \{T_s\sigma_s \mid 1 \leq s \leq p, T_s \in \Sigma^+, \sigma_s \in \mathcal{T} \subset \mathcal{S}_n\}$$

is a set of potential solutions of SCOS for $\mathbf{S}$. We consider $\mathbf{T}$ as a set of individual chromosomes. Let $q(T_s\sigma_s, \mathbf{S})$ is the largest number such that $S_{\sigma_s(i)}$ is a subsequence of T_s, for all $1 \leq i \leq q(T_s\sigma_s, \mathbf{S})$, and there are

$$F_{S_{\sigma_s(1)}}, F_{S_{\sigma_s(2)}}, \ldots, F_{S_{\sigma_s(q(T_s\sigma_s, \mathbf{S}))}}$$

such that $F_{S_{\sigma_s(i)}}(1) < F_{S_{\sigma_s(j)}}(1)$ and $F_{S_{\sigma_s(i)}}(|S_{\sigma_s(i)}|) < F_{S_{\sigma_s(j)}}(|S_{\sigma_s(j)}|)$, for all $i < j$ where $i, j \in \{1, 2, \ldots, q(T_s\sigma_s, \mathbf{S})\}$. We use

$$\frac{q(T_s\sigma_s, \mathbf{S})}{|T_s|}$$

as the fitness function.

During each successive generation, a proportion P of the existing population is selected to breed a new generation. We assume that $P = \lceil \frac{|\mathbf{T}|}{2} \rceil$.

Chromosomes are selected by the fitness function. We can use two genetic operators, crossover, mutation. For GA[1], we consider only crossover. Usually, crossover defines a part of parent chromosome which used for construction of child chromosome. We use standard random crossover.

We use a simple genetic algorithm GA[2] to evolve a set of function

$$\mathbf{F} = \{\sum_{\beta,\gamma} \alpha(\beta,\gamma) x^{\beta} y^{\gamma} \mid \alpha(\beta,\gamma) \in \mathcal{Q}, \beta \in \mathcal{Z}, \gamma \in \mathcal{Z}\}.$$

We can consider functions from $\mathbf{F}$ as fitness functions for GA[1]. In particular, we assume that $x = |T_s|$ and $y = q(T_s\sigma_s, \mathbf{S})$. We use the rate of convergence of GA[1] with fitness function $F \in \mathbf{F}$ as the fitness function of GA[2]. GA[1] with evolved fitness function we denote by GA[3].

Let $r(T_s\sigma_s, \mathbf{S})$ is the smallest number such that $S_{\sigma_s(i)}$ is a subsequence of T_s, for all $r(T_s\sigma_s, \mathbf{S}) \leq i \leq n$, and there are

$$F_{S_{\sigma_s(r(T_s\sigma_s,\mathbf{S}))}}, F_{S_{\sigma_s(r(T_s\sigma_s,\mathbf{S})+1)}}, \ldots, F_{S_{\sigma_s(n)}}$$

such that $F_{S_{\sigma_s(i)}}(1) < F_{S_{\sigma_s(j)}}(1)$ and $F_{S_{\sigma_s(i)}}(|S_{\sigma_s(i)}|) < F_{S_{\sigma_s(j)}}(|S_{\sigma_s(j)}|)$, for all $i < j$ where $i, j \in \{r(T_s\sigma_s, \mathbf{S}), r(T_s\sigma_s, \mathbf{S}) + 1, \ldots, n\}$. It is clear that we can assume that $x = |T_s|$, $y = n - r(T_s\sigma_s, \mathbf{S})$ and consider functions from $\mathbf{F}$ as fitness functions for GA[1]. We use the rate of convergence of GA[1] with fitness function $F \in \mathbf{F}$ as the fitness function of GA[2]. GA[1] with such evolved fitness function we denote by GA[4].

We consider a genetic algorithm GA[5] for coevolution of populations of GA[3] and GA[4]. Let $\mathbf{T}_{1,1}, \mathbf{T}_{1,2}, \ldots, \mathbf{T}_{1,p[1]}$ be a set of populations of $p[1]$ parallel launched genetic algorithms GA[3]. Let $\mathbf{T}_{2,1}, \mathbf{T}_{2,2}, \ldots, \mathbf{T}_{2,p[2]}$ be a set of populations of $p[2]$ parallel launched genetic algorithms GA[4]. Let

$$\mathbf{T}_{i,j} = \{T_{i,j,s}\sigma_{i,j,s} \mid 1 \leq s \leq p, T_{i,j,s} \in \Sigma^{+}, \sigma_{i,j,s} \in \mathcal{T}_{i,j} \subset \mathcal{S}_n\}.$$

Let

$$\begin{aligned}\mathbf{C} = \{(q[0], q[1], \ldots, q[9]) \quad \mid \quad & 0 \leq q[0] \leq 1, 1 \leq q[1] \leq p[1], 1 \leq q[2] \leq p[2], \\ & 0 \leq q[3] \leq 1, 0 \leq q[4] \leq 1, 0 \leq q[5] \leq 1, \\ & 1 \leq q[6] \leq p[1], 1 \leq q[7] \leq p[2], \\ & 0 \leq q[8] \leq 1, 0 \leq q[9] \leq 1, \\ & q[0] \in \mathcal{Q}, q[1] \in \mathcal{N}, q[2] \in \mathcal{N}, q[3] \in \mathcal{Q}, q[4] \in \mathcal{Q}, \\ & q[5] \in \mathcal{Q}, q[6] \in \mathcal{N}, q[7] \in \mathcal{N}, q[8] \in \mathcal{Q}, q[9] \in \mathcal{Q}\}\end{aligned}$$

is a set of individual chromosomes of GA[5]. We assume that $q[0]$ is current fitness of the chromosome. During each successive generation, we select chromosomes

	10^3	10^4	10^5	10^6
GA[1]	3.26	2.18	1.63	1.58
GA[3] (10^2)	2.41	1.74	1.37	1.33
GA[3] (10^4)	2.17	1.53	1.24	1.19
GA[5] (10^2, 0)	1.39	1.14	1.07	1.066
GA[5] (10^4, 0)	1.17	1.08	1.039	1.0372
GA[5] (10^2, 10^3)	1.072	1.044	1.0273	1.0108
GA[5] (10^4, 10^3)	1.053	1.028	1.0145	1.00354

Table 2: Experimental results for different genetic algorithms and numbers of generations.

$$T_{1,q[1],\lceil q[3]|\mathbf{T}_{1,q[1]}|\rceil}, T_{2,q[2],\lceil q[4]|\mathbf{T}_{1,q[2]}|\rceil}.$$

Value of $q[5]$ defines proportions of partitions of selected chromosomes. Two new chromosomes we use to replace chromosomes

$$T_{1,q[6],\lceil q[8]|\mathbf{T}_{1,q[6]}|\rceil}, T_{2,q[7],\lceil q[9]|\mathbf{T}_{1,q[7]}|\rceil}.$$

If fitness of child chromosomes greater than fitness of parent chromosomes, then $q[0] + \frac{1-q[0]}{2}$ is a new value of fitness of $(q[0], q[1], \ldots, q[9])$. If fitness of parent chromosomes greater than fitness of child chromosomes, then $0.9q[0]$ is a new value of fitness of $(q[0], q[1], \ldots, q[9])$. During new run, GA[5] use final population from the previous run.

Note that we can obtain optimal solutions for SCOS using satisfiability algorithms. This optimal solutions we use to run GA[5] in test mode. During test mode, we compare solutions of GA[5] and optimal solutions. If GA[5] gives us a good solution, then $q[0]+\frac{1-q[0]}{2}$ is a new value of fitness of $(q[0], q[1], \ldots, q[9])$, for any chromosome used in the run. Otherwise, $0.9q[0]$ is a new value of fitness of $(q[0], q[1], \ldots, q[9])$, for any chromosome used in the run.

Let T is a solution of some genetic algorithm. Let T_{opt} is an optimal solution. The value of $\frac{|T|}{|T_{opt}|}$ we can use to rate the quality of the genetic algorithm. Let GA$[i](g)$ is GA$[i]$ with the fitness function after g generations of GA[2] where $i \in \{3, 4\}$. Let GA$[5](g, t)$ is GA[5] with GA$[3](g)$ and GA$[4](g)$ after t rounds of test mode. Selected experimental results for different genetic algorithms are given in Tab. 2.

ACKNOWLEDGEMENTS. The work was partially supported by Analytical Departmental Program "Developing the scientific potential of high school", RFBR, research project No. 13-01-00048 a, and Ural Federal University development program with the financial support of young scientists.

References

[1] V. Yu. Popov, Computational complexity of problems related to DNA sequencing by hybridization, *Doklady Mathematics*, 72 (2005), 642-644.

[2] V. Popov, The approximate period problem for DNA alphabet, *Theoretical Computer Science*, 304 (2003), 443-447.

[3] V. Popov, The Approximate Period Problem, *IAENG International Journal of Computer Science*, 36 (2009), 268-274.

[4] V. Popov, Approximate Periods of Strings for Absolute Distances, *Applied Mathematical Sciences*, 6 (2012), 6713-6717.

[5] V. Popov, Multiple genome rearrangement by swaps and by element duplications, *Theoretical Computer Science*, 385 (2007), 115-126.

[6] V. Popov, Sorting by prefix reversals, *IAENG International Journal of Applied Mathematics*, 40 (2010), 247-250.

[7] K. Räihä and E. Ukkonen, The shortest common supersequence problem over a binary alphabet is NP-complete, *Theoretical Computer Science*, 16 (1981), 187-198.

[8] A. Gorbenko and V. Popov, The Problem of Finding Two Edge-Disjoint Hamiltonian Cycles, *Applied Mathematical Sciences*, 6 (2012), 6563-6566.

[9] A. Gorbenko and V. Popov, Footstep Planning for Humanoid Robots, *Applied Mathematical Sciences*, 6 (2012), 6567-6571.

[10] A. Gorbenko and V. Popov, Hamiltonian Alternating Cycles with Fixed Number of Color Appearances, *Applied Mathematical Sciences*, 6 (2012), 6729-6731.

[11] A. Gorbenko and V. Popov, Task-resource Scheduling Problem, *International Journal of Automation and Computing*, 9 (2012), 429-441.

[12] A. Gorbenko and V. Popov, Longest Common Parameterized Subsequences with Parameterized Common Substring, *Applied Mathematical Sciences*, 7 (2013), 2341-2345.

[13] A. Gorbenko and V. Popov, The Minimum k-Cover Problem, *Applied Mathematical Sciences*, 7 (2013), 2347-2352.

[14] A. Gorbenko and V. Popov, The Shortest Common Superstring Problem, *Applied Mathematical Sciences*, 7 (2013), 2353-2356.

[15] A. Gorbenko and V. Popov, Computational Experiments for the Problem of Footstep Planning for Humanoid Robots, *Applied Mathematical Sciences*, 7 (2013), 2357-2372.

[16] A. Gorbenko and V. Popov, The Shortest Common Parameterized Supersequence Problem, *Applied Mathematical Sciences*, 7 (2013), 2373-2380.

[17] A. Gorbenko and V. Popov, On the Problem of Placement of Visual Landmarks, *Applied Mathematical Sciences*, 6 (2012), 689-696.

[18] A. Gorbenko and V. Popov, The set of parameterized k-covers problem, *Theoretical Computer Science*, 423 (2012), 19-24.

Received: June 15, 2013